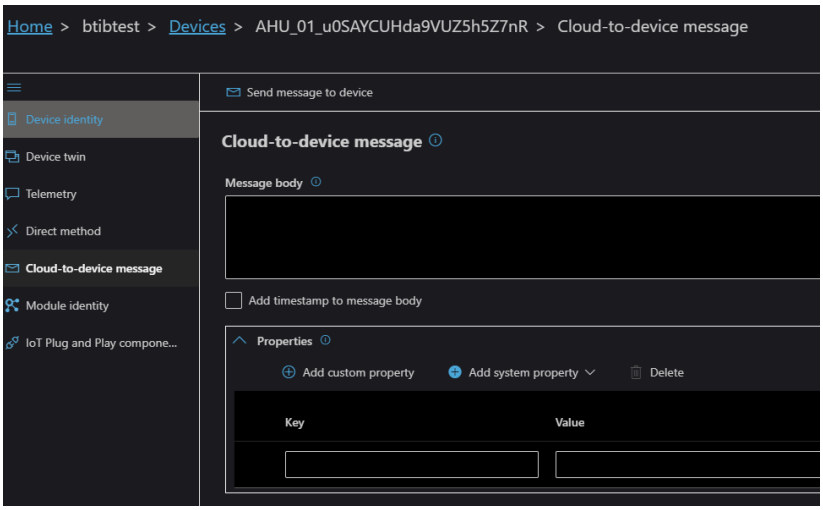
Step 5 Send commands from IoTHub to Niagara

IoTHub Connector supports cloud to niagara communication via IoTHub C2D messages to control points remotely to do this.

Send a point action command

1. Go to the **Azure IoT Explorer** application then in your IoTHub devices then click on the device where you want to send messages. Tehn cloud to device messages.



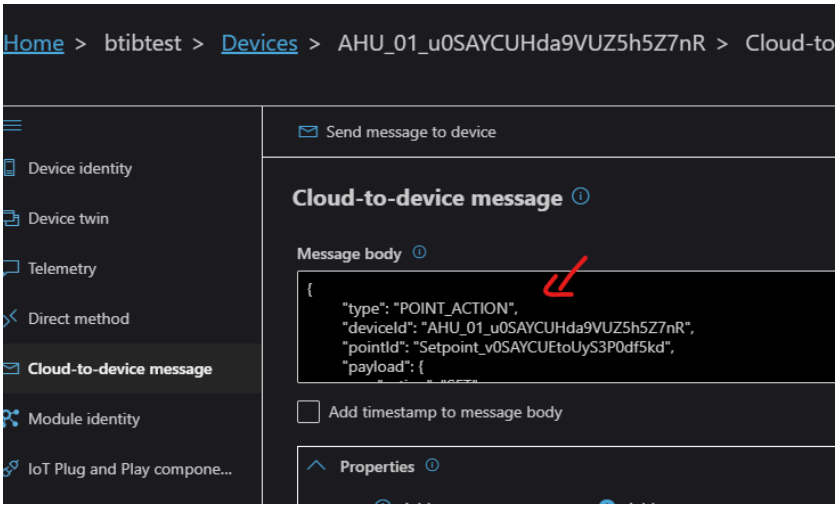
2. By default we use this message template for **POINT_ACTION** command. by you can use any format that meat your needs. check the connector advanced settings.
    a. This is the default command template.

```
{
        "type": "POINT_ACTION",
        "deviceId": "AHU_01_u0SAYCUHda9VUZ5h5Z7nR",
        "pointId": "Setpoint_v0SAYCUEtoUyS3P0df5kd",
        "payload": {
                "action": "SET",
                "value": 1111
        }
}
```

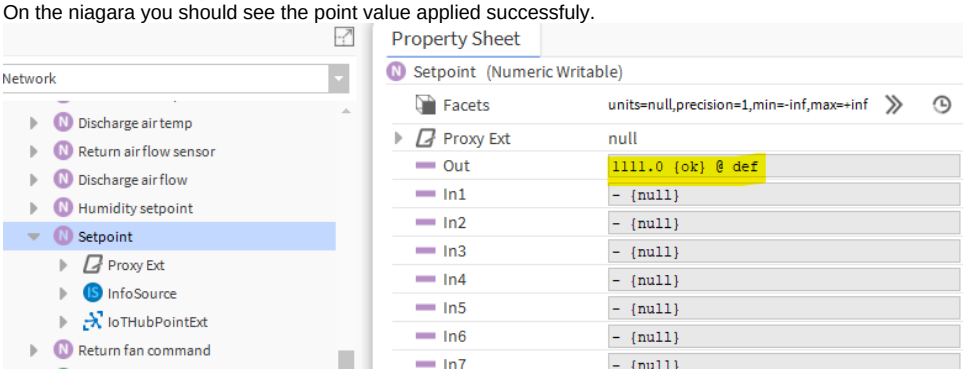
    b. This is the default connector point action command configuration.

Commands Policy	Single Point Command
Message Type	{json('type')}
Command Set Object	{json('').escape}
Command Device Id	{json('deviceId')}
Command Point Id	{json('pointId')}
Command Action	{json('payload.action')}
Command Value	{json('payload.value')}
Command Duration	{json('payload.duration')}

3. On the left chose Cloud To Device Message.



4. Then hit **Send Message**.
5. On the niagara you should see the point value applied successfuly.

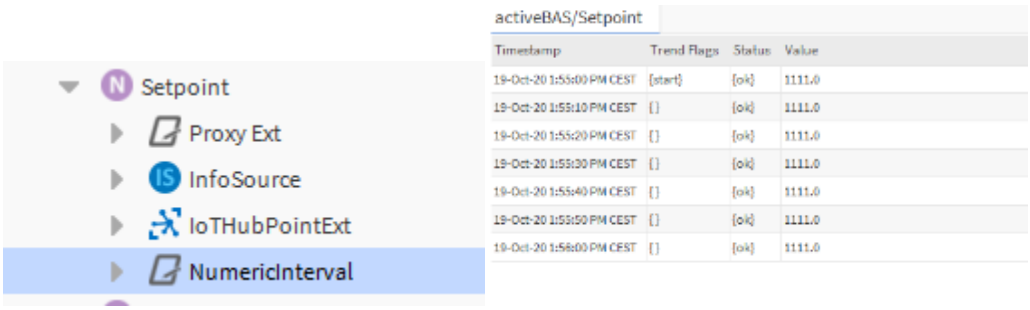


6. And the new value sent to the cloud.

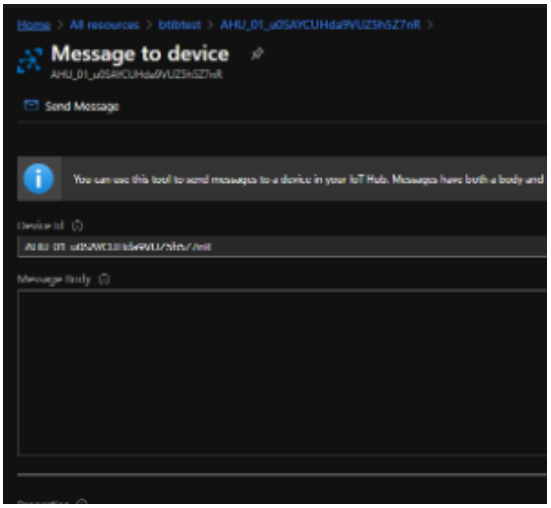
Send a send history command

You can pull historical value for any point that has a history associated by sending a **SEND_HISTORY** command.

1. Add a history. extension to the point.



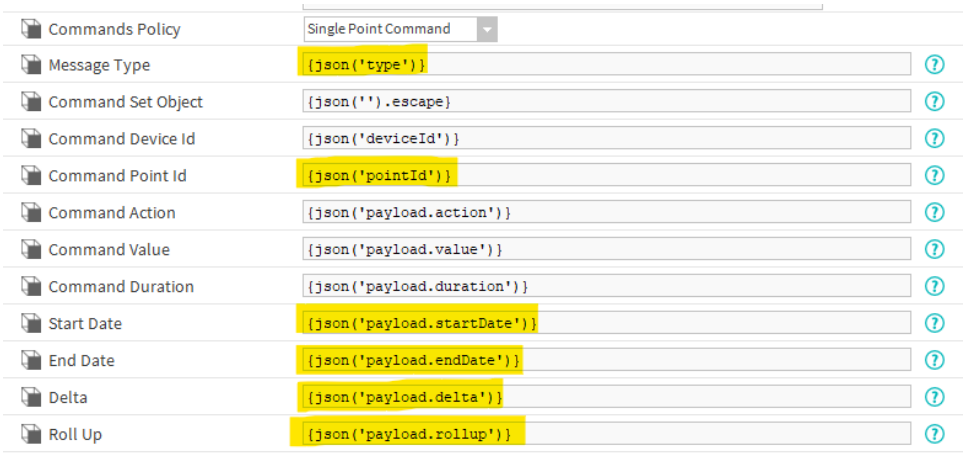
2. Go to the portal and send the command to the device.



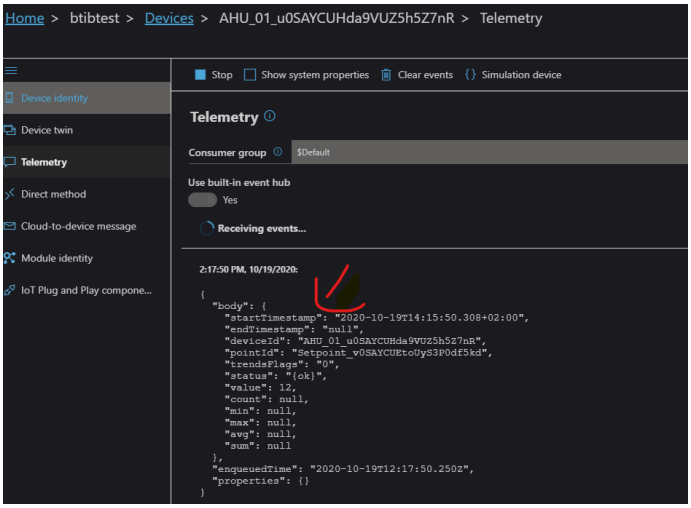
3. By default we use this message template for send history command. by you can use any format that meat your needs. check the connector advanced settings.
    a. This is the default command template.

```
{
        "type": "SEND_HISTORY",
        "deviceId": "AHU_01_u0SAYCUHda9VUZ5h5Z7nR",
        "pointId": "Setpoint_v0SAYCUEtoUyS3P0df5kd",
        "payload": {
                "startDate": "2020-10-19T13:15:55.011+02:00",
                "endDate": "2020-10-19T14:15:55.011+02:00"
        }
}
```

    b. This is the default connector send history command configuration.



4. And you should see the messages being sent.



To change the message format check the connector advanced setting then history message template

History Message Variables	$(startTimestamp) $(endTimestamp) $(pointId) $(deviceId) $(status) $(value) $(trendsFlags) $(count) $(min) $(max) $(avg) $(sum)
History Message Template	{ "startTimestamp": "$(startTimestamp)", "endTimestamp": "$(endTimestamp)", "deviceId": "$(deviceId)", "pointId": "$(pointId)", "trendsFlags": "$(trendsFlags)", "status": "$(status)", "value" : $(value), "count": $(count), "min": $(min),

Send ack alarm command

You can ack alarms by sending an **ACK_ALARM** command to any alarm recipient device.

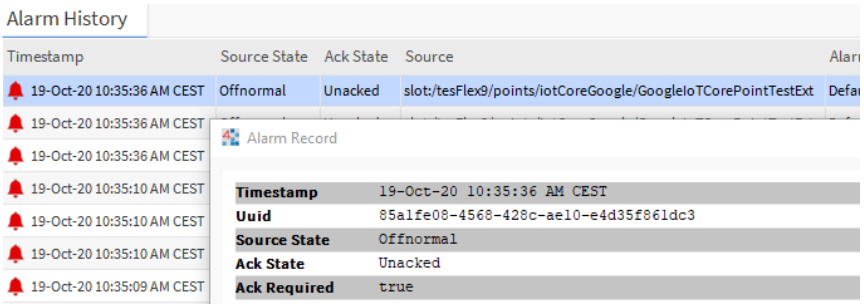
1. By default we use this message template for ack alarm command. by you can use any format that meat your needs. check the connector advanced settings.
    a. This is the default command template.

```
{
        "type": "ACK_ALARM",
        "payload": {
                "uuid": "85a1fe08-4568-428c-ae10-e4d35f861dc3"
        }
}
```

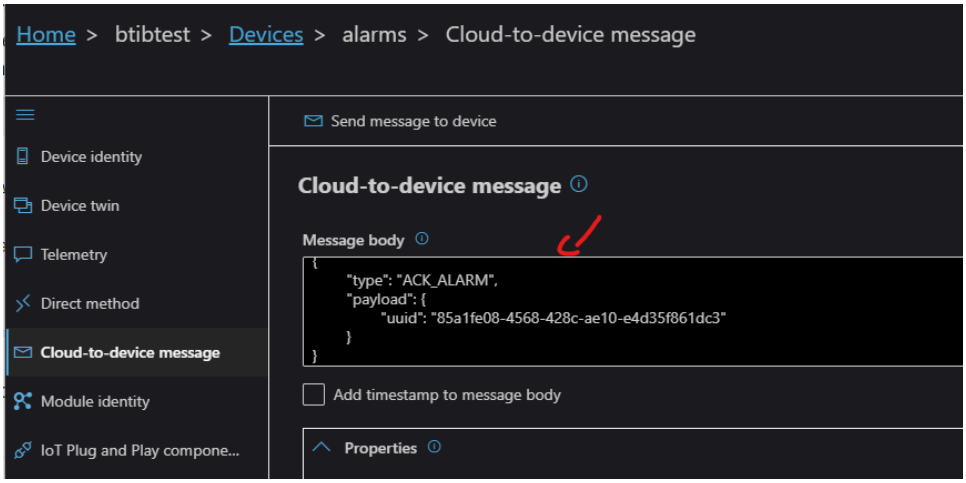
    b. This is the default connector send history command configuration.

	"min": $(min),
Alarm Uuid	{json('payload.uuid')}

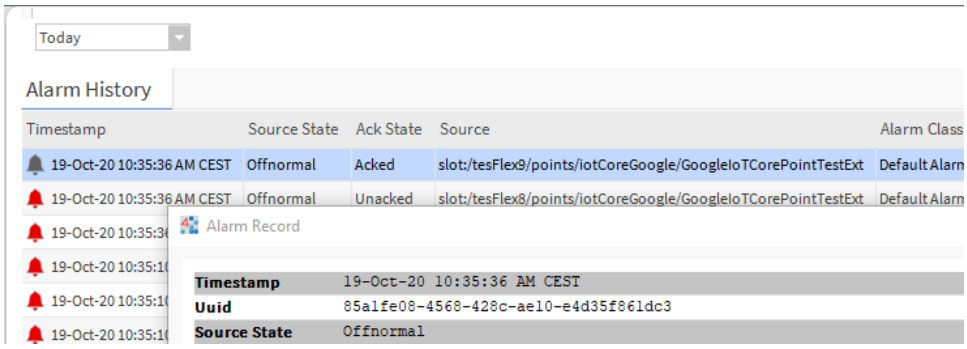
2. Go the alarms console and pick an unacked alarm id.



3. Go to device and send the command.



4. Go back to the console and you should see that the alarm has been acked.



Next Step

Step 6 Consume Data from IoTHub in Azure